Mini Project Assignment: Daily Calorie Tracker CLI

Course: Programming for Problem Solving using Python

Assignment Title: Building a Calorie Tracking Console App

Assignment Type: Individual

Estimated Duration: 8-10 hours

Weightage: 15% of course grade

Real-World Problem Context

Many students want a quick and simple way to monitor their daily calorie intake. This mini project aims to help them build a Python-based CLI (Command Line Interface) tool where they can log their meals and keep track of total calories consumed, compare against a personal daily limit, and save session logs for future tracking.

Learning Objectives

By completing this mini project, learners will:

- Use input(), type conversion, and basic I/O operations in Python.
- Store and manage user input using lists and basic data structures.
- Apply arithmetic and comparison operators for calculations and conditions.
- Practice string formatting using f-strings and escape characters.
- Learn basic file I/O to save session logs.
- Strengthen logical thinking by building a real-world CLI tool.

Assignment Tasks

Task 1: Setup & Introduction

- Create a project folder daily_calorie_tracker/
- Inside the folder, create a starter Python script tracker.py
- Add a comment header with your name, date, and project title.
- Print a welcome message describing what the tool does.

Deliverable: tracker.py script with welcome message and program intro.

Task 2: Input & Data Collection

- Use input() to accept:
    - Meal name (e.g., “Breakfast”)
    - Calorie amount (e.g., “350”) — convert it to int or float
- Store the meal names in one list and calorie amounts in another list.
- Ask the user how many meals they want to enter (loop accordingly).

Deliverable: Working meal & calorie input logic, with list storage.

Task 3: Calorie Calculations

- Use sum() to calculate total calorie intake.
- Compute average calorie per meal.
- Use arithmetic operators to perform the calculations.
- Ask user to input their daily calorie limit and compare it to total.

Deliverable: Code for total, average calorie calculation and comparison.

Task 4: Exceed Limit Warning System

- Use if...else or if statements with comparison operators:
    - If calorie intake > daily limit → show a warning message.
    - Else → show a success/within-limit message.

Deliverable: Warning system logic using comparison operators.

Task 5: Neatly Formatted Output

- Print a summary table with the following using f-strings:

Meal Name Calories

Breakfast 350

Lunch 600

Snack 150

Total: 1100

Average: 366.67

- Use tabs (\t), newlines (\n), and proper alignment for neatness.

Deliverable: Well-formatted CLI report printed to screen.

Task 6 (Bonus): Save Session Log to File

- Ask user if they want to save the report.
- If yes, use open("filename.txt", "w") to write session data to a file:
    - Include timestamp, meal details, total & average calories, limit status.

Deliverable: A .txt file containing session summary.

Submission Instructions

- tracker.py Python script with all tasks implemented
- At least 3 sample runs with different input values
- calorie_log.txt file (if bonus task is completed)
- Header comments and user-friendly prompts
- Neat code formatting and appropriate use of comments
- Submit the GitHub repository URL via LMS

Evaluation Rubric

Criterion	Weight	Excellent (5)	Good (4)	Fair (3)	Poor (2)
Git-based project setup	10%	Reproducible, structured	Functional	Incomplete	Not attempted
CRISP-DM problem framing	20%	Insightful, clear metrics	Adequate	Vague	Missing
Data acquisition & documentation	15%	Source & metadata logged	Partial	Lacks clarity	Missing
Data audit & profiling	20%	Clean visual report, good summary	Basic stats	Missing insights	Not attempted
Data cleaning & transformation	25%	Well-handled, saved & commented	Partial	Limited	Not done
Code quality, readability, and ethics	10%	Clean, modular, no plagiarism	OK	Sloppy	Copied or missing

Academic Integrity & Plagiarism Policy

- Submissions must be original and individual.
- Copied code, notebooks, or canvases will result in zero credit and disciplinary action.
- You may use public tutorials or documentation for reference but must cite them in your README.md.

Deadline: 7 days
Contact for Queries: jyoti.yadav@krmangalam.edu.in

Happy coding and stay curious!